4

InnerTube: скачиваем, конвертируем и синхронизируем видеоролики с YouTube

Автор	Дэн Фернандес
Сложность	Высокая
Необходимое время	20+ часов
Стоимость	бесплатно
Программное обеспечение	Visual C# 2008 Express с Service Pack 1, Visual Basic 2008 Express Edition с Service Pack 1, iTunes версии 7.0 или выше
Оборудование	Необязательно: устройство iPod, iPhone или Zune для синхронизации
Адрес в Интернете	*http://www.codeplex.com/InnerTube/* и *http://www.c4fbook.com/InnerTube*

Признаюсь, я люблю YouTube. Это единственный сайт в Интернете, где найдутся видеоматериалы на любой вкус: от фрагментов классических фильмов типа «Криминальное чтиво» или «Большой Лебовски» до таких интернет-знаменитостей, как Трон Гай (Tron Guy), Кимбо Слайс (Kimbo Slice) или Тай Зондай (Tay Zonday) (он же Mr. Chocolate Rain). Можно сидеть и смотреть бесконечно.

Короче говоря, на сайте YouTube есть все, кроме транспортабельности. Увы, смотреть видео с YouTube можно только при наличии соединения с Интернетом. Цель проекта InnerTube – снять это досадное ограничение. Мы научимся автоматически скачивать видео с YouTube,

конвертировать их в более подходящий формат и даже копировать на устройство iPod или Zune.

Краткий обзор

Чтобы понять, на что способна программа InnerTube, вкратце рассмотрим ее основные функции.

Пользовательский интерфейс InnerTube

На рис. 4.1 видно, что главное окно InnerTube состоит из трех панелей и по структуре аналогично Outlook. В левой панели отображается перечень лент, в средней – список видеороликов в выбранной ленте, а в правой – детальная информация о выбранном ролике.

Рис. 4.1. *Окно программы InnerTube*

Первый запуск InnerTube

При первом запуске InnerTube проверяет, установлено ли на вашем компьютере клиентское программное обеспечение iTunes или Zune и предлагает указать папку, в которой будут храниться скачанные ролики (услужливо предлагая вариант по умолчанию). Если ни iTunes, ни Zune не обнаружены, то функция синхронизации с ними отключается (см. рис. 4.2).

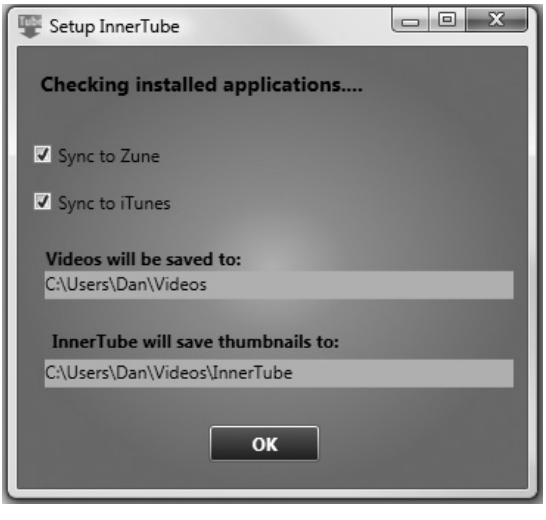

Рис. 4.2. *InnerTube конфигурирует себя для работы на вашем ПК*

Добавление видео

Выбрав из главного меню пункт Add Videos, вы сможете без труда добавить различные типы лент с YouTube (рис. 4.3). Для каждой ленты можно задать различные свойства, например: имя пользователя, избранные ролики которого вас интересуют, или интервал времени между скачиваниями роликов с наивысшим рейтингом (Top Rated Over Time).

Как работает InnerTube

Познакомившись с тем, что программа InnerTube делает, рассмотрим, как она устроена, а именно:

- принцип работы InnerTube API;
- классы InnerTube для работы с роликами и лентами;

Рис. 4.3. *Добавление лент в InnerTube*

- преобразование XML-документов, возвращаемых YouTube API, в объекты InnerTube;
- порядок загрузки видеороликов с YouTube;
- конвертирование видеоформатов;
- краткий обзор фоновых процессов InnerTube;
- синхронизация с iTunes и Zune;
- построение приложения InnerTube;
- конструирование пользовательского интерфейса InnerTube, основанного на технологии WPF.

Принцип работы YouTube API

YouTube API представляет собой REST-службу, в которой используется формат Google Gdata, позволяющий выполнять такие операции, как поиск и извлечение информации о видеороликах, а также загрузку роликов на сайт YouTube.

В отличие от других служб, например Live.com или Amazon.com, не требуется заводить учетную запись разработчика и получать ключ API, чтобы читать данные с YouTube. На рис. 4.4 показаны результаты запроса роликов с самым высоким рейтингом за все время существования YouTube, полученные с помощью броузера Firefox.

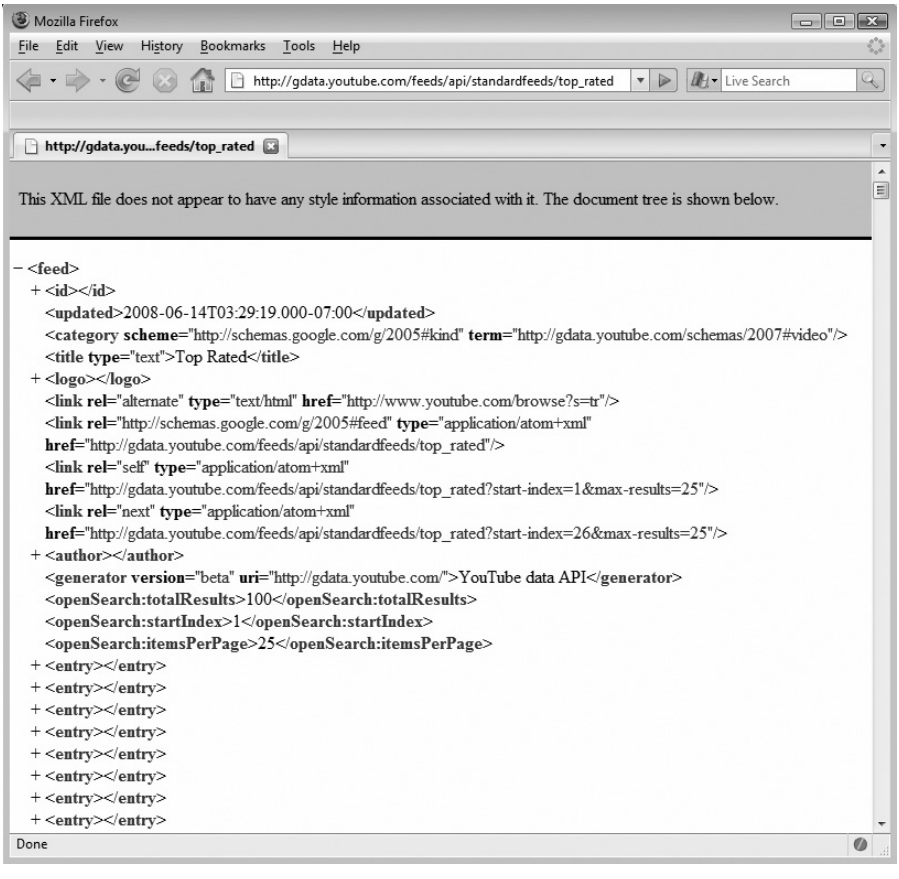

Рис. 4.4. *XML-документ, возвращаемый YouTube API*

Примечание

Полная документация о службах, предоставляемых YouTube, находится на странице *http://code.google.com/apis/youtube/developers_guide_protocol.html*.

Стандартные ленты YouTube

Приглядевшись к URL на рис. 4.4, вы заметите, что лента Top Rated относится к категории стандартных – «Standard Feed». На сайте YouTube определено несколько стандартных лент, в том числе:

- Top Rated (с наивысшим рейтингом): *http://gdata.youtube.com/feeds/api/standardfeeds/top_rated*
- Top Favorited (чаще всего помещаемые к папку «Избранное»): *http://gdata.youtube.com/feeds/api/standardfeeds/top_favorites*

- Most Viewed (чаще всего просматриваемые): *http://gdata.youtube.com/feeds/api/standardfeeds/most_viewed*

Вы можете ограничить множество роликов, извлекаемых из стандартной ленты, добавив в строку запроса параметр `time`. Например, в ответ на следующий запрос будут возвращены только ролики, которые чаще всего просматривались на этой неделе:

```
http://gdata.youtube.com/feeds/api/standardfeeds/most_viewed?time=this_week
```

Параметр `time` может принимать следующие значения: `today`, `this_week`, `this_month`, `all_time`. Если он не задан, то по умолчанию подразумевается `all_time` (за все время).

Связанные с пользователями ленты YouTube

По умолчанию YouTube раскрывает данные о пользователе (например, ваши избранные ролики и подписки). Они доступны с помощью API. Чтобы получить избранное или подписки некоторого пользователя, достаточно вместо строки *username* подставить в приведенный ниже URL имя существующего пользователя YouTube. Но в отличие от стандартных лент, вы не можете включить в такой URL параметр `time` и тем самым профильтровать связанную с пользователем ленту по времени.

- Избранное указанного пользователя: *http://gdata.youtube.com/feeds/api/users/username/favorites*
- Подписки указанного пользователя *http://gdata.youtube.com/feeds/api/users/username/ subscriptions*

Поиск по сайту YouTube

YouTube предоставляет также API поиска. Для обращения к нему следует включить в URL параметр vq=*searchterm*. Например, следующий URL описывает запрос на поиск всех роликов Microsoft:

```
http://gdata.youtube.com/feeds/api/videos?vq=Microsoft
```

Дополнительные параметры в строке запроса

Обертка, которую InnerTube реализует вокруг YouTube, поддерживает только простой поиск, но интерфейс самого YouTube включает также ряд сервисных механизмов, в частности разбиение на страницы и сортировку. Чтобы ими воспользоваться, необходимо включить в строку запроса дополнительные параметры. Ниже перечислены некоторые из них:

`orderby`

Задает порядок сортировки результатов, например: по релевантности, по рейтингу, по дате публикации или по количеству просмотров. По умолчанию подразумевается сортировка по релевантности.

`max-results`

Задает максимальное количество возвращаемых результатов. По умолчанию равно 25.

`start-index`

Режим разбиения на страницы. Параметр `start-index` должен быть равен порядковому номеру результата, с которого должна начинаться выборка. Если `start-index` равно 100, то API вернет результаты с 100 по 125. По умолчанию равен 0.

Примечание

Полный перечень параметров в строке запроса см. на странице *http://code.google.com/apis/youtube/reference.html#Query_parameter_definitions*.

Еще некоторые важные URL на сайте YouTube

На сайте YouTube есть ряд других проверенных URL, которые следует знать при создании приложения. Приведем их краткие описания:

Просмотр

Это ссылка, которая ведет непосредственно на видеоролик: *http://www.youtube.com/watch?v=VideoID*

Встраивание

Эта ссылка предназначена для размещения ролика на другом сайте, например MySpace: *http://www.youtube.com/v/VideoID*

Миниатюра

Это ссылка на миниатюру видео максимально возможного размера (425×344): *http://www.youtube.com/v/VideoID*

Скачивание

Это скрытая ссылка, которой мы будем пользоваться для скачивания ролика: *http://www.youtube.com/get_video?video_id=VideoID&t=sessiontoken*

Примечание

Ссылка для скачивания ролика будет работать, только если включить в URL корректный маркер сеанса, зависящий от времени. Это сделано для того, чтобы не дать пользователям скачивать ролики напрямую. Мы покажем, как обойти это ограничение, в разделе «Скачивание роликов с YouTube» ниже.

Классы для работы с роликами и лентами YouTube

Теперь, когда вы познакомились с устройством YouTube API, покажем, как представить типы возвращаемых API данных в виде классов .NET (рис. 4.5).

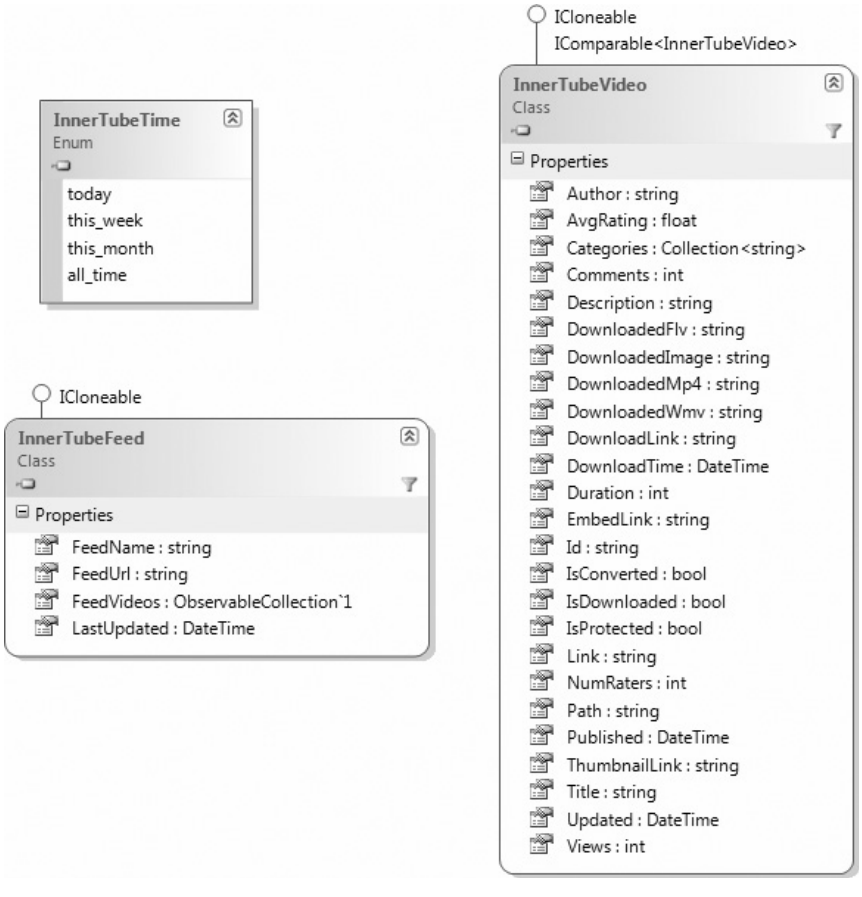

***Рис. 4.5.** Отображение YouTube API на классы .NET*

`InnerTubeTime`

Это перечисление представляет все возможные значения параметра `time`. Мы воспользуемся им в классе `InnerTubeService` для описания необязательного параметра, передаваемого YouTube API.

`InnerTubeFeed`

Этот класс представляет одну конкретную ленту, например Top Rated. Содержит понятное пользователю имя ленты `Feed Name` и URL, указывающий на один из перечисленных выше компонентов YouTube API.

`InnerTubeVideo`

Представляет один ролик YouTube и включает большую часть полей, возвращаемых YouTube (автор, комментарии, категории), а также свойства, относящиеся к уже скачанному ролику, например его местоположение на диске.

Строки или класс Uri?

Возможно, вы обратили внимание на то, что в классе `InnerTubeVideo` URL хранятся в виде строк, а не объектов класса Uri. Вообще говоря, для этой цели лучше использовать класс Uri, но InnerTube должна сериализовывать (уметь сохранять) свои классы на диск с помощью класса XmlSerializer. К сожалению, класс Uri не является сериализуемым и возбуждает исключение при попытке сериализации. Связано это с тем, что в классе Uri не определен конструктор без параметров.

Вызов класса InnerTubeService

Спроектировав классы, покажем, как с их помощью можно преобразовать XML-документ, изображенный на рис. 4.4, в набор объектов, представленных на рис. 4.5. Для этого воспользуемся еще одним классом: `InnerTubeService`. На рис. 4.6 видно, что стандартным лентам – Top

Рис. 4.6. *IntelliSense показывает методы для получения данных с YouTube*

Rated, Most Viewed и т. д. – соответствуют методы этого класса. Все они возвращают объект типа `ObservableCollection<InnerTubeVideo>`.

Зачем нужен класс ObservableCollection?

Сразу стоит отметить, что наборы видеороликов представляются классом `ObservableCollection`. Этот класс очень похож на `Collection`, но дополнительно реализует два события: `CollectionChanged` и `PropertyChanged`, которые упрощают обновление связанных элементов данных в WPF. Вопрос о связывании мы подробно обсудим в разделе «Анатомия MainWindow.xaml» ниже, а пока скажем лишь, что этот тип дает возможность автоматически обновлять элементы интерфейса при изменении набора, не требуя повторной привязки к данным.

Преобразование YouTube XML-документов в объекты .NET

По сути дела, в классе `InnerTubeService` есть всего один метод, который занимается преобразованием XML-документа в набор объектов, – `ConvertYouTubeXmlIntoObjects`. Объясняется это тем, что все возвращаемые YouTube API документы имеют одинаковую структуру, будь то результаты поиска или список роликов с наивысшим рейтингом. Поэтому и код для разбора любого XML-документа будет один и тот же, хотя мы все же включили ряд вспомогательных методов, например `Search(string query)`, чтобы упростить работу с библиотекой.

Из примеров 4.1 и 4.2 видно, что метод `ConvertYouTubeXmlToObjects` принимает два параметра: `Uri` некоторого компонента YouTube API и `Setting` – объект, описывающий конфигурацию каталогов на вашем ПК, в которых будут сохраняться различные файлы (видео, миниатюры и т. д.). Класс `Setting` мы подробнее рассмотрим ниже в разделе «Настройки приложения».

Пример 4.1. Сигнатура метода ConvertYouTubeXmlToObjects (версия на C#)

```
public static ObservableCollection<InnerTubeVideo>
ConvertYouTubeXmlToObjects(
  Uri youTubeUrl, Setting setting)
```

Пример 4.2. Сигнатура метода ConvertYouTubeXmlToObjects (версия на VB)

```
Public Shared Function ConvertYouTubeXmlToObjects(ByVal youTubeUrl As Uri, _
  ByVal setting As Setting) As ObservableCollection(Of InnerTubeVideo)
```

В теле метода `ConvertYouTubeXmlToObjects` мы объявляем несколько переменных типа `XNamespace`. Это необходимо, поскольку в получаемом от

YouTube документе широко используются пространства имен XML (см. примеры 4.3 и 4.4).

Пример 4.3. Объявление пространств имен XML (версия на C#)

```
XNamespace nsBase = @"http://www.w3.org/2005/Atom";
XNamespace nsGData = @"http://schemas.google.com/g/2005";
XNamespace nsYouTube = @"http://gdata.youtube.com/schemas/2007";
```

Пример 4.4. Объявление пространств имен XML (версия на VB)

```
Dim nsBase As XNamespace = "http://www.w3.org/2005/Atom"
Dim nsGData As XNamespace = "http://schemas.google.com/g/2005"
Dim nsYouTube As XNamespace = "http://gdata.youtube.com/schemas/2007"
```

Чтобы получить сами данные по сети, мы вызываем службу YouTube, пользуясь классом `WebClient`, который находится в пространстве имен `System.Net`. Мы обращаемся к методу `OpenRead()` этого класса, передавая ему URL службы. Результат будет точно такой же, как при получении XML-документа в броузере (см. рис. 4.5). Затем с помощью объекта `XmlTextReader` мы загружаем документ в объект типа `XDocument` (примеры 4.5 и 4.6).

Пример 4.5. Отправка веб-запроса YouTube API (версия на C#)

```
// Необходим для обращения к службе
WebClient wc = new WebClient();

// Получаем данные
XmlTextReader xr = new XmlTextReader(wc.OpenRead(youTubeUrl));
XDocument rawData = XDocument.Load(xr);
```

Пример 4.6. Отправка веб-запроса YouTube API (версия на VB)

```
' Необходим для обращения к службе
Dim wc As New WebClient()

' Получаем данные
Dim xr As New XmlTextReader(wc.OpenRead(youTubeUrl))
Dim rawData As XDocument = XDocument.Load(xr)
```

В этот момент мы имеем в памяти XML-документ, содержащий список видеороликов. Каждый элемент `<entry>` (один ролик) представлен в нем объектом типа `XElement`. Теперь мы хотим использовать технологию *LINQ for XML* для перебора всех элементов `<entry>` и создания объекта `InnerTubeVideo`, в котором будут установлены свойства, описывающие автора, категории, название и т. д. (см. примеры 4.7 и 4.8).

В первой строке в обоих примерах обратите внимание на то, что в качестве параметра методу `Descendents` передается `nsBase + "entry"`. Это необходимо, так как элемент `<entry>` находится в пространстве имен `nsBase` и без указания этого пространства имен LINQ-запрос возбудил бы исключение с сообщением о том, что такого атрибута у элемента не существует. Отметим также, что мы не строки конкатенируем, а используем

перегруженный оператор + в классе XNamespace. Именно поэтому в версии на Visual Basic используется оператор +, а не оператор конкатенации строк (&).

Пример 4.7. Цикл перебора элементов <entry> и установки свойств (версия на C#)

```
var query = from entry in rawData.Descendants(nsBase + "entry")
  select new InnerTubeVideo
  {
    Author = entry.Element(nsBase + "author").Element(nsBase + "name").Value,
    Categories = ParseCategories(entry.Elements(nsBase + "category")),
    Id = ParseID(entry.Element(nsBase + "id").Value),
    Published = DateTime.Parse(entry.Element(nsBase + "published").Value),
    Updated = DateTime.Parse(entry.Element(nsBase + "updated").Value),
    Title = entry.Element(nsBase + "title").Value,
    Description = entry.Element(nsBase + "content").Value,
...
```

Пример 4.8. Цикл перебора элементов <entry> и установки свойств (версия на VB)

```
Dim query = From entry In rawData.Descendants(nsBase + "entry") _
  Select New InnerTubeVideo With _
  {
    .Author = entry.Element(nsBase + "author").Element(nsBase +
              "name").Value, _
    .Categories = ParseCategories(entry.Elements(nsBase + "category")), _
    .Id = ParseID(entry.Element(nsBase + "id").Value), _
    .Published = DateTime.Parse(entry.Element(nsBase + "published").Value), _
    .Updated = DateTime.Parse(entry.Element(nsBase + "updated").Value), _
    .Title = entry.Element(nsBase + "title").Value, _
    .Description = entry.Element(nsBase + "content").Value, _...
```

Выше уже отмечалось, что некоторые интересующие нас поля находятся в различных пространствах имен, поэтому в программе нам приходится указывать подходящее пространство имен. Чтобы было понятнее, о чем идет речь, в примере 4.9 приведен фрагмент исходного XML-документа, полученного от YouTube API (содержит общее количество просмотров ролика).

Пример 4.9. Фрагмент XML-документа, содержащий общее количество просмотров ролика

```
<yt:statistics viewCount="130534" favoriteCount="1210" />
```

Поскольку атрибут viewCount находится в пространстве имен <yt> (сокращение от YouTube), то для получения его значения мы должны явно добавить переменную nsYouTube, в которой хранится ссылка на это пространство имен. В примерах 4.10 и 4.11 мы начинаем с корневого элемента entry, получаем вложенный в него элемент statistics и затем читаем атрибут viewCount.

Пример 4.10. Получение атрибута viewCount в XML-документе (версия на C#)

```
Views = int.Parse(entry.Element(nsYouTube +
          "statistics").Attribute("viewCount").Value),
```

Пример 4.11. Получение атрибута viewCount в XML-документе (версия на VB)

```
.Views = Integer.Parse(entry.Element(nsYouTube + _
          "statistics").Attribute("viewCount").Value),
```

У *LINQ for XML* есть удобная возможность, позволяющая сделать разбор XML-документа относительно компактным. Речь идет о передаче функции порции XML. Например, сногсшибательный ролик Джонни Ли (*http://www.youtube.com/watch?v=cI1AwZN4ZYg*) входит в 15 разных категорий YouTube, как показано в примере 4.12.

Пример 4.12. Фрагмент XML-документа, содержащий категории видеороликов

```
<category scheme="…" term="wiimote" />
<category scheme="…" term="display" />
<category scheme="…" term="3D" />
<category scheme="…" term="reality" />
<category scheme="…" term="nintendo" />
...
```

Мы хотим преобразовать атрибуты `term` элементов `<category>` в удобную для работы переменную типа `Collection<string>`, в которой каждый `term` будет представлен строкой `string`. Для этого мы вызываем метод `ParseCategories`, как показано в примерах 4.13 и 4.14.

Пример 4.13. Передача фрагмента XML методу ParseCategories (версия на C#)

```
Categories = ParseCategories(entry.Elements(nsBase + "category")),
```

Пример 4.14. Передача фрагмента XML методу ParseCategories (версия на VB)

```
.Categories = ParseCategories(entry.Elements(nsBase + "category")),
```

Метод `ParseCategories` получает на входе объект типа `IEnumerable<Xelement>`, который отформатирован точно так же, как элементы `<category>` в примере 4.12. Теперь мы можем воспользоваться LINQ, чтобы найти в каждом элементе `category` атрибут `term` и вернуть их в виде объекта `Collection<string>`, как показано в примерах 4.15 и 4.16.

Пример 4.15. Извлечение атрибутов term (версия на C#)

```
private static Collection<string> ParseCategories(IEnumerable<XElement>
        Categories)
{
  var vals = from c in Categories.Attributes("term")
```

```
    select c.Value;
  return (Collection<string>)vals;
}
```

Пример 4.16. Извлечение атрибутов term (версия на VB)

```
Private Shared Function ParseCategories(ByVal Categories As
        IEnumerable(Of XElement))
As Collection(Of String)
  Dim vals = From c In Categories.Attributes("term") _
  Select c.Value
  Return vals.ToCollection()
End Function
```

В примерах 4.17 и 4.18 приведен полный текст метода `ConvertYouTube-XmlToObjects`.

Пример 4.17. Метод ConvertYouTubeXmlToObjects (версия на C#)

```
public static ObservableCollection<InnerTubeVideo>
ConvertYouTubeXmlToObjects(
  Uri youTubeUrl, Setting setting)
{
  XNamespace nsBase = @"http://www.w3.org/2005/Atom";
  XNamespace nsGData = @"http://schemas.google.com/g/2005";
  XNamespace nsYouTube = @"http://gdata.youtube.com/schemas/2007";

  // Использовать для обращения к службе
  WebClient wc = new WebClient();

  // Получить данные
  XmlTextReader xr = new XmlTextReader(wc.OpenRead(youTubeUrl));
  XDocument rawData = XDocument.Load(xr);
  var query = from entry in rawData.Descendants(nsBase + "entry")
  select new InnerTubeVideo
  {
    Author = entry.Element(nsBase + "author").Element(nsBase + "name").Value,
    Categories = ParseCategories(entry.Elements(nsBase + "category")),
    Id = ParseID(entry.Element(nsBase + "id").Value),
    Published = DateTime.Parse(entry.Element(nsBase + "published").Value),
    Updated = DateTime.Parse(entry.Element(nsBase + "updated").Value),
    Title = entry.Element(nsBase + "title").Value,
    Description = entry.Element(nsBase + "content").Value,
    ThumbnailLink = _BaseThumbnailUrl + ParseID(entry.Element(nsBase +
                    "id").Value) + @"/0.jpg",
    Link = _BasewatchUrl + ParseID(entry.Element(nsBase + "id").Value),
    EmbedLink = _baseEmbedUrl + ParseID(entry.Element(nsBase + "id").Value),
    DownloadLink = _BaseDownloadUrl + ParseID(entry.Element(nsBase +
       "id").Value),
    Views = int.Parse(entry.Element(nsYouTube +
       "statistics").Attribute("viewCount").Value),
    AvgRating = float.Parse(entry.Element(nsGData +
       "rating").Attribute("average").Value),
    NumRaters = int.Parse(entry.Element(nsGData +
```

```
        "rating").Attribute("numRaters").Value),

      // установить, где сохранять скачанные файлы
      DownloadedImage = FileHelper.BuildFileName(setting.SubPath,
        ParseID(entry.Element(nsBase + "id").Value), FileType.Image),
      DownloadedFlv = FileHelper.BuildFileName(setting.SubPath,
        entry.Element(nsBase + "title").Value, FileType.Flv),
      DownloadedMp4 = FileHelper.BuildFileName(setting.VideoPath,
        entry.Element(nsBase + "title").Value, FileType.Mp4),
      DownloadedWmv = FileHelper.BuildFileName(setting.VideoPath,
        entry.Element(nsBase + "title").Value, FileType.Wmv)
    };

    return query.ToObservableCollection<InnerTubeVideo>();
  }
```

Пример 4.18. Метод ConvertYouTubeXmlToObjects (версия на VB)

```
Public Shared Function ConvertYouTubeXmlToObjects(ByVal youTubeUrl As Uri, _
  ByVal setting As Setting) As ObservableCollection(Of InnerTubeVideo)

  Dim nsBase As XNamespace = "http://www.w3.org/2005/Atom"
  Dim nsGData As XNamespace = "http://schemas.google.com/g/2005"
  Dim nsYouTube As XNamespace = "http://gdata.youtube.com/schemas/2007"

  ' Использовать для обращения к службе
  Dim wc As New WebClient()

  ' Получить данные
  Dim xr As New XmlTextReader(wc.OpenRead(youTubeUrl))
  Dim rawData As XDocument = XDocument.Load(xr)
  Dim query = From entry In rawData.Descendants(nsBase + "entry") _

  Select New InnerTubeVideo With _
  { _
    .Author = entry.Element(nsBase + "author").Element(nsBase +
        "name").Value, _
    .Categories = ParseCategories(entry.Elements(nsBase + "category")), _
    .Id = ParseID(entry.Element(nsBase + "id").Value), _
    .Published = DateTime.Parse(entry.Element(nsBase + "published").Value), _
    .Updated = DateTime.Parse(entry.Element(nsBase + "updated").Value), _
    .Title = entry.Element(nsBase + "title").Value, _
    .Description = entry.Element(nsBase + "content").Value, _
    .ThumbnailLink = _BaseThumbnailUrl & ParseID(entry.Element(nsBase + _
      "id").Value)& "/0.jpg", _
    .Link = _BasewatchUrl & ParseID(entry.Element(nsBase + "id").Value), _
    .EmbedLink = _baseEmbedUrl & ParseID(entry.Element(nsBase + "id").Value), _
    .DownloadLink = _BaseDownloadUrl & ParseID(entry.Element(nsBase +
        "id").Value), _
    .Views = Integer.Parse(entry.Element(nsYouTube + _
        "statistics").Attribute("viewCount").Value), _
    .AvgRating = Single.Parse(entry.Element(nsGData + _
        "rating").Attribute("average").Value), _
    .NumRaters = Integer.Parse(entry.Element(nsGData + _
      "rating").Attribute("numRaters").Value), _
```

```
            .DownloadedImage = FileHelper.BuildFileName(setting.SubPath, _
              ParseID(entry.Element(nsBase + "id").Value), FileType.Image), _
            .DownloadedFlv = FileHelper.BuildFileName(setting.SubPath,
              entry.Element(nsBase + "title").Value, FileType.Flv), _
            .DownloadedMp4 = FileHelper.BuildFileName(setting.VideoPath, _
              entry.Element(nsBase + "title").Value, FileType.Mp4), _
            .DownloadedWmv = FileHelper.BuildFileName(setting.VideoPath, _
              entry.Element(nsBase + "title").Value, FileType.Wmv) _
        }

        Return query.ToObservableCollection()

    End Function
```

Теперь, когда мы научились получать информацию о размещенных на YouTube роликах, посмотрим, как можно скачать ролик из программы.

Скачивание видеороликов с YouTube

Сейчас мы покажем, как скачивать с YouTube видеоролики – сначала вручную с помощью броузера, а потом и программно.

Скачивание ролика с помощью броузера

На сайте YouTube видеролики хранятся в формате Flash video (FLV). Именно в таком формате мы и будем их получать.

Но прежде чем программировать скачивание, посмотрим, как это можно сделать с помощью броузера. Для начала откройте любую страницу YouTube с роликом, например знаменитый ролик Тэя Зондея «Chocolate Rain» (Шоколадный дождь) (*http://www.youtube.com/watch?v=EwTZ2xpQwpA*). Из этого URL следует, что идентификатор ролика (параметр v) равен «EwTZ2xpQwpA».

YouTube вас не пускает (в соответствии с проектным решением)

Существует (скрытый) URL, позволяющий скачать видео напрямую, но в нем нужно указать два параметра: `video_id` и *маркер сеанса*. Маркер сеанса – это идентификатор, который YouTube назначает вашему броузеру; он действителен в течение примерно 15 минут. Если у вас нет действующего маркера сеанса, то YouTube отклонит запрос на скачивание Flash-ролика.

Чтобы убедиться в этом, откройте свой броузер и перейдите на страницу *http://www.youtube.com/get_video?video_id=EwTZ2xpQwpA*. Смотрите-ка – YouTube притворяется, будто получил негодный URL, он возвращает броузеру Internet Explorer ответ `HTTP 404 Not Found` (Страница не найдена) (пользователи Firefox 3 увидят пустую страницу) (рис. 4.7). Все дело в том, что в URL следует включить действительный маркер сеанса.